XT3/4 Architecture and Software

NCCS USERS MEETING

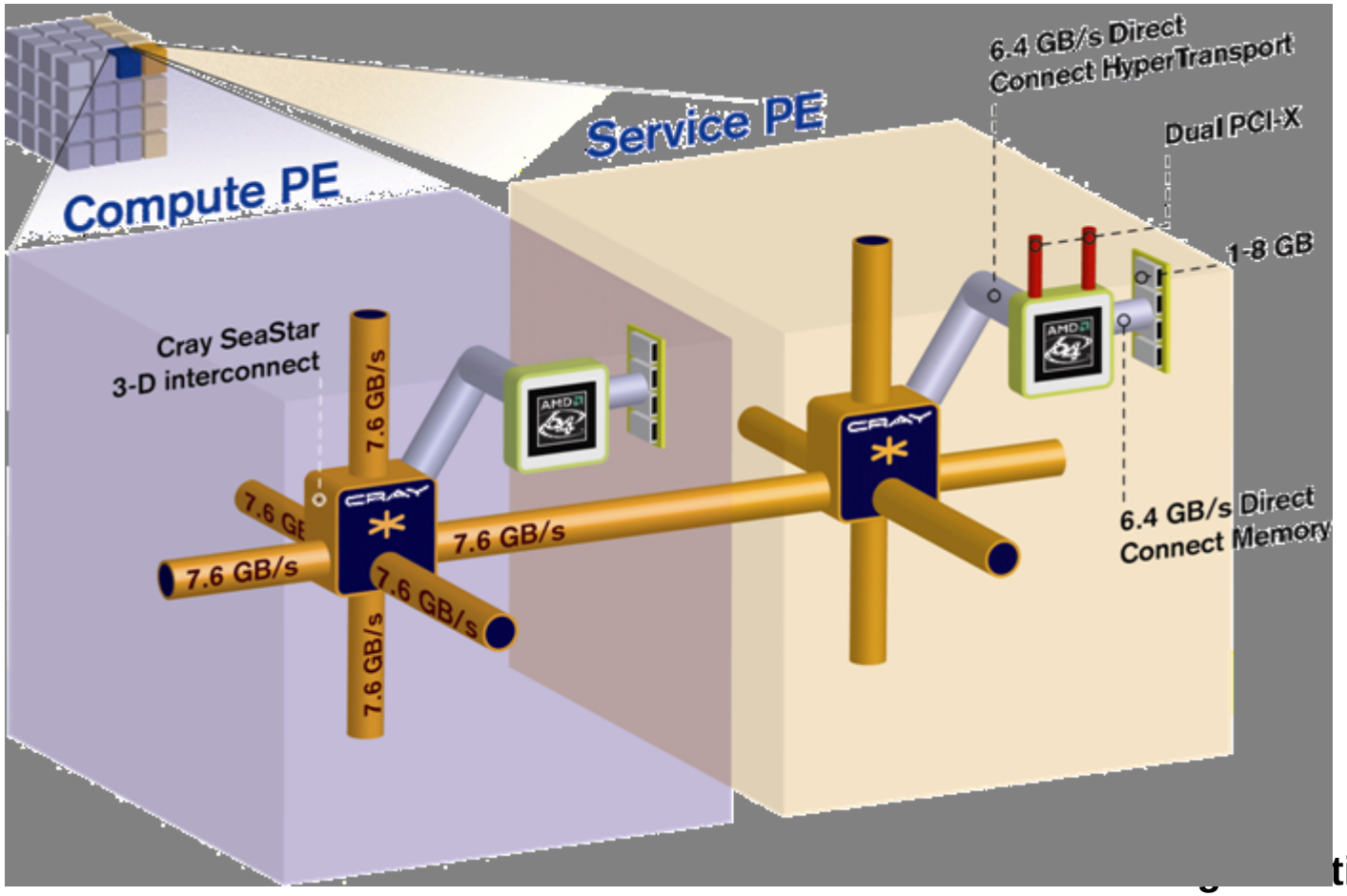
Ramanan Sankaran
Scientific Computing Group

"jaguar" is a combination of XT3 and XT4

	XT3	XT4	Total
No. of cabinets	56	68	124
No. of compute sockets	5212	6296	11508

- **Each compute socket has a 2.6GHz dual-core AMD opteron processor**
  - 11508 x 2 = 23016 cores
- **Memory is 4GB/processor or 2GB/core**
  - 11508 x 4GB = 45TB
- **jaguar's aggregate peak performance is ~119TF**
  - 10.4GF per socket
- **In addition to compute nodes, there are service nodes for I/O, login etc.**

Cray XT3/4 Architecture



tion Cray MPP

- **Service nodes run Linux**
- **Compute nodes run Catamount quintessential kernel (qk)**

Getting started on jaguar...

- **Connecting**
  - **`ssh <your_username>@jaguar.ccs.ornl.gov`**
- **File systems**
  - Home directory is /spin/home/<*username*>
    - Accessible from all NCCS systems
    - Regularly backed-up
    - Quotas exist. Use **`lsquota`** to check usage
  - Scratch space is `/tmp/work/<username>`
    - Points to the lustre file system
    - Not backed up. Periodically purged
      - Files not accessed in more than a week are eligible for purging

Current software environment

- **PGI 6.1.6**
- **gcc 3.3**
- **Login nodes have kernel 2.6.5**
- **XT/MPT 1.5.31**
- **acml 3.6**

Customizable through modules

modules

- **Several software available as modules**
- **`module {list/avail/load/unload}`**
- **`module swap` worth remembering**
- **Watch for the occasional information message when executing `module load`**

```
% module load netcdf
Usage: ftn test.f90 ${NETCDF_F_LIB}   or  cc test.c
  ${NETCDF_C_LIB}
```

What is different under catamount?

- No threading (pthreads or OpenMP)
- No TCP/IP facilities (pipes, sockets or IP messages)
- No popen(), fork(), exec() or system() calls
- No dynamic (shared) libraries. static linking is the only option
- The /proc file-system is not available
- No IPC calls (shared memory shmem, limited signal handling).
- No mmap(), sbrk()
- No profil()
- No etime(), times(), clock()
- Limited ioctl()
- No terminal control
- No unix style deamons supported functions

Compilers

- **`ftn`, `cc`, and `CC` are very tidy wrappers for catamount compiling & linking.**
- **Use the wrappers essentially all the time.**
  - most of your builds will be cross-compiles for catamount

```
/opt/xt-pe/1.5.31/bin/snos64/ftn: INFO: catamount target is being used
```

`-target=catamount` will suppress litany of warnings

- **`-r8` to do ubiquitous scientific computing promotion**
- **`-g` to get debugging symbols**
  - put `-g` FIRST (it implies `-O0`)
  - `-Ktrap=fp` to trap floating point exceptions, and thereby actually do useful debugging

Compiler options for optimization

- **-fast to optimize**

```
% pgf90 -fast -help
-fast  Common optimizations: -O2 -Munroll=c:1 -Mnoframe -Mlre
```

- **Try some vectorization with -fastsse**
  - Only buys you 1 extra flop/clock for `REAL*8`, but fewer instructions are generated

```
-fastsse == -fast -Mvect=sse -Mscalarsse -Mcache_align -Mflushz
```

  - `-Mcache_align`: if you don't use `-fastsse` to build main, makes sure arrays are on cache line boundaries

Compiler options (cont...)

- **Let the compiler unroll small loops**
  - e.g. -Munroll=c:4 unrolls loops 4 times
- **`-tp k8-64` explicitly sets optimization for 64-bit Opteron**
- **`-Mipa=fast` enables interprocedural analysis (IPA)**
  - Equivalent to Mipa=align,arg,const,f90ptr, shape, globals,localarg,ptr
  - It is usually a good thing for C++
  - Make sure to put it on the link line too
- **`-byteswapio` for big-endian data format**

Compiler optimization report

- **`-Minfo=all` emits information, including whether SSE instructions were generated**
  - same as -Minfo=inline,ipa,loop,mp

Sample output from compiling with `-fastsse -Minfo=all`

```
step_icd2:
   205, Generated 4 alternate loops for the inner loop
        Generated vector sse code for inner loop
        Generated 2 prefetch instructions for this loop
        Generated vector sse code for inner loop
        Generated 2 prefetch instructions for this loop
        Generated vector sse code for inner loop
        Generated 2 prefetch instructions for this loop
        Generated vector sse code for inner loop
        Generated 2 prefetch instructions for this loop
        Generated vector sse code for inner loop
        Generated 2 prefetch instructions for this loop
   225, Loop unrolled 4 times
```

Running on jaguar

- **Queue management and scheduling is done through torque and moab**
  - torque is based on PBS
- **A sample job script**

```
#!/bin/csh
#PBS -A XXXYYY
#PBS -N test
#PBS -joe
#PBS -lwalltime=1:00:00,size=1024
#PBS -W depend=afterany:<jobid>
#PBS -l feature=xt4

set_environment_variables_here
executable_part_of_batch_script
```

csh will be used to interpret the script
A project code is necessary

size is the number of 'sockets' requested
Introduce a job dependency (optional)
Choose to run on xt3 or xt4 (optional)

Running (cont...)

- **By default commands will be executed on the service nodes**
- **yod launches applications on compute nodes**

  yod -size <size> -SN/VN executable

  – -SN executes on only one core per socket

  – -VN executes on both cores of a socket (default)

```
#!/bin/csh
#PBS -A XXXYYY
#PBS -N test
#PBS -joe
#PBS -lwalltime=1:00:00,size=1024

...
yod -size 1024 -SN ./a.out
yod -size 2048 -VN ./a.out
```

csh will be used to interpret the script
A project code is necessary

size is the number of 'sockets' requested

Uses all sockets, 1 core per socket
Uses all sockets, both cores per socket

yod and small_pages

- **`-small_pages` option to `yod`**
  - Opteron TLB provides 512 entries for 4kB pages, or 8 entries for 2MB pages.
  - By default, Catamount uses 2MB pages
  - This allows 16MB to be mapped in the TLB (vs 2MB for 4kB pages)
  - If your code jumps around to more than 8 places in memory (e.g. you have some sort of gather/scatter loop), you may want to try `-small_pages`

Useful MPI variables

- **You may need to (re)set a couple of MPI environment variables**
- **`MPICH_UNEX_BUFFER_SIZE` - size of buffers for unexpected receives**
  - Default = 60M
  - >400M?
- **`MPICH_PTL_OTHER_EVENTS` - sets the number of events in queue to receive "all other" types of messages (i.e. a lot, e.g. MPI_ALL_TO_ALL)**
  - Default = 2048
  - 4096 works for some codes to go to 5000 procs

More MPI variables

- **`MPICH_PTL_UNEX_EVENTS` - number of unexpected point-to-point messages (MPI_GATHERV)**
  - Default = 20480
  - Experience shows may need to be set to 80000 or more
- **`MPICH_RANK_REORDER_METHOD` - controls the assignment of MPI ranks**
  - Set to 1 for smp-style (0,1;2,3;4,5)
  - Set to 2 for folded (0,3;1,4;2,5)
  - Set to 3 for custom. You must then create a file in your run directory named MPICH_RANK_ORDER. This file is a comma separated (ranges allowed) list of ranks

Submitting and monitoring jobs

- **Submit a job using `qsub <batch_script>`**
- **`qstat -a` shows the queue status**
- **`qstat -u <username>` shows the users' jobs**
- **`qalter` can change some job characteristics**
- **The Moab utility `showq` can be used to view a more detailed description of the queue**
  - Shows the state of the job. Active, idle, blocked etc.
  - Shows the priority of different jobs in the queue
- **checkjob and showstart are other useful Moab utilities**
  - Show why a job is blocked, expected start time etc.

More monitoring tools

- **Watch your job with `xtshowmesh` or `xtshowcabs`**
- **yod may die during start-up or in-between due to hardware failure**
  - Can your application restart using checkpoints?
  - Have multiple `yod` in the batch script with `sleep` in between
  - If one yod crashes, the next yod can start within the same batch job
- **If you should need to kill a `yod`**
  - `xtps -Y` to find out the nid and pid
  - `xtkill -9 <nid>.<pid>` deletes the yod without removing the job

Queue policies

- **Two queues : production and debug**
  - `#PBS -q batch` or `#PBS -q debug` in batch script
- **~10% of the machine is reserved for the debug queue from 10am-10pm, Mon-Fri.**
  - Only one debug job at a time
  - Maximum wall-time of 1 hour
- **Batch jobs have time limits depending on job size**

  < 128 : Max. 4 hours

  129 - 2000 : Max 12 hours

  > 2000 : Max 24 hours
- **Only two jobs per user will be in 'eligible' state. Rest will be in 'blocked' state**
  - Jobs that are running are not counted in the 'two jobs'.

Interactive debugging

- **Interactive jobs are useful for debugging**

```
% qsub -I -V -qdebug -A<XXXYYY> -lWalltime=1:00:00,size=32
qsub: waiting for job 9493.jaguar10.ccs.ornl.gov to start
qsub: job 9493.jaguar10.ccs.ornl.gov ready

% cd to_the_right_path
% yod -np 64 -VN ./executable.x
```

- **Totalview is available on jaguar**

```
% totalview yod -np 64 -VN ./executable.x
```

- **Debug queue is to be used for software development, testing and debugging only**
- **Do not use it for production work**

Accounting

- **Hours charged = job_size x 2 x walltime**
  - Jobs are allocated an entire socket and not individual cores
  - A job will be charged for both cores irrespective of whether one or both cores in a socket are used
  - XT3 and XT4, both are charged same
- **showusage is useful to track account usage**

```
% showusage
 Usage on jaguar:
                        Project Totals      <userid>
 Project     Allocation      Usage   Remaining       Usage
_____________________________|_____________________________|_____________
 <YourProj>   2000000  |  123456.78  1876543.22 |   1560.80

%
```

Scientific Libraries

- **ACML (AMD Core Math Library)**
  - BLAS, LAPACK, 1-D FFT
  - Fast intrinsics and vector intrinsics
  - LAPACK timing routines have been hacked
  - Has been compiled with `-fastsse`, so use `-Mcache_align`
- **Cray LibSci**
  - ScaLAPACK, BLACS, SuperLU
- **`acml/3.6` and `xt-libsci/1.5.31` are loaded as part of the default module set**
- **fftw/2.1.5 and fftw/3.1 are available**

I/O Libraries

- **HDF5**
  - Parallel and serial versions available as modules (`hdf5/1.6.4_ser` & `hdf5/1.6.4_par`)
  - Need to add link and include info to build
    - `${HDF5_FLIB}` and `${HDF5_CLIB}`
    - These also point to szip and libz
- **netCDF**
  - netcdf/3.6.0 available as module

    ```
    ftn test.f90 ${NETCDF_F_LIB} or
    cc test.c ${NETCDF_C_LIB}
    ```

  - Any need for pnetCDF?
- **Please let us know what other libraries you need**

lustre filesystem